PyPalEx 2.2.0

Generated by Doxygen 1.9.8

1 PyPalEx: The Python Palette Extractor	1
1.1 Description	 . 1
2 Namespace Index	1
2.1 Package List	 . 1
3 Class Index	2
3.1 Class List	 . 2
4 File Index	2
4.1 File List	 . 2
5 Namespace Documentation	3
5.1 pypalex Namespace Reference	 . 3
5.1.1 Detailed Description	 . 3
5.2 pypalexmain Namespace Reference	 . 3
5.2.1 Function Documentation	 . 5
5.2.2 Variable Documentation	 . 7
5.3 pypalex.arg_messages Namespace Reference	 . 10
5.3.1 Function Documentation	 . 10
5.4 pypalex.constants Namespace Reference	 . 11
5.4.1 Variable Documentation	 . 12
5.5 pypalex.conversion_utils Namespace Reference	 . 18
5.5.1 Function Documentation	 . 19
5.6 pypalex.extraction_utils Namespace Reference	 . 23
5.6.1 Function Documentation	 . 24
5.7 pypalex.Extractor Namespace Reference	 . 32
5.8 pypalex.file_utils Namespace Reference	 . 32
5.8.1 Function Documentation	 . 32
5.9 pypalex.image_utils Namespace Reference	 . 34
5.9.1 Function Documentation	 . 34
5.10 pypalex.print_utils Namespace Reference	 . 36
5.10.1 Function Documentation	 . 37
6 Class Documentation	41
6.1 Extractor Class Reference	 . 41
6.1.1 Detailed Description	 . 42
6.1.2 Constructor & Destructor Documentation	 . 43
6.1.3 Member Function Documentation	 . 43
6.1.4 Member Data Documentation	 . 49
7 File Documentation	50
7.1mainpy File Reference	 . 50
7.1.1 Detailed Description	52

7.1.2 Author(s)	52
7.2 arg_messages.py File Reference	52
7.2.1 Detailed Description	53
7.2.2 Author(s)	53
7.3 constants.py File Reference	53
7.3.1 Detailed Description	55
7.3.2 Author(s)	55
7.4 conversion_utils.py File Reference	55
7.4.1 Detailed Description	56
7.4.2 Author(s)	56
7.5 extraction_utils.py File Reference	56
7.5.1 Detailed Description	57
7.5.2 Author(s)	58
7.6 Extractor.py File Reference	58
7.6.1 Detailed Description	58
7.6.2 Author(s)	58
7.7 file_utils.py File Reference	59
7.7.1 Detailed Description	59
7.7.2 Author(s)	59
7.8 image_utils.py File Reference	59
7.8.1 Detailed Description	60
7.8.2 Author(s)	60
7.9 print_utils.py File Reference	60
7.9.1 Detailed Description	61
7.9.2 Author(s)	61
Index	63

# 1 PyPalEx: The Python Palette Extractor

# 1.1 Description

PyPalEx is a tool for extracting color palettes from images and generating a JSON format file with light and dark color themes. This tool is intended to be OS independent, for use by the tech community for developing their own custom theme managers or by artists who want to extract color palettes for their art from images, pictures or wallpapers they adore.

# 2 Namespace Index

# 2.1 Package List

Here are the packages with brief descriptions (if available):

nymelov	
pypalex Python Palette Extractor: extracts color palettes from images	3
pypalexmain	3
pypalex.arg_messages	10
pypalex.conversion_utils	
pypalex.conversion_utils	18
pypalex.extraction_utils	23
pypalex.Extractor  pypalex.file_utils	
pypalex.print_utils	36
3 Class Index	
3.1 Class List	
Here are the classes, structs, unions and interfaces with brief descriptions:	
Extractor Extracts colors given a matrix of HSV values extracted from an image	41
4 File Index	
4.1 File List	
Here is a list of all files with brief descriptions:	
mainpy	50
Main script for PyPalEx	50
arg_messages.py Archive of messages to display for arguments supplied by user	52
constants.py A collection of constants for PyPalEx	53
conversion_utils.py	
Utilities for converting between RGB, HSV, HEX	55
extraction_utils.py Utilities for extracting colors from the image	56
Extractor.py Extraction utility class for extracting colors from the image	58

file_utils.py Utilities for file handling	59
image_utils.py Utilities for processing image and file handling	59
print_utils.py Utilities for printing preview to the screen	60

# 5 Namespace Documentation

# 5.1 pypalex Namespace Reference

Python Palette Extractor: extracts color palettes from images.

#### **Namespaces**

- namespace \_\_main\_\_
- namespace arg\_messages
- namespace constants
- · namespace conversion utils
- namespace extraction\_utils
- namespace Extractor
- namespace file\_utils
- namespace image\_utils
- namespace print utils

### 5.1.1 Detailed Description

Python Palette Extractor: extracts color palettes from images.

PyPalEx is a tool for extracting color palettes from images and generating a JSON format file with light and dark color themes. This tool is intended to be OS independent, for use by the tech community for developing their own custom theme managers or by artists who want to extract color palettes for their art from images, pictures or wallpapers they adore.

### 5.2 pypalex.\_\_main\_\_ Namespace Reference

## **Functions**

• main ()

Main script function.

• handle\_args ()

Handles the arguments passed to PyPalEx.

extract\_color\_palettes ()

Handles color extraction from image(s).

• setup argument parser ()

Sets up the argument parser for command line arguments.

check\_sources (filepaths, path=None)

Checks each of the sources provided and removes any bad sources.

check\_path (path)

Check the path to make sure it exists.

• handle\_config ()

Handle the PyPalEx configuration file settings.

set\_global\_args (args)

Sets the global variables using the arguments.

preview\_and\_save (extractor, save\_type, img\_index)

Shows a preview of and saves the extracted color palette(s).

check\_source (filepath)

Checks to make sure the path leads to a file.

#### **Variables**

• str CONFIG FILENAME = 'palex-config.yaml'

Filename of the configuration file.

• list PROPER\_IMAGES = []

List of real/existing image file path(s).

• list FILENAMES = []

List of image filenames (contain file extensions).

• list IMAGE\_NAMES = []

List of image names.

• str OUTPUT PATH = "

The path to the output directory where all exported files will be saved.

str EXPORT\_FILE\_FORMAT = 'json'

The format of the files to be exported (e.g.

• str EXPORT\_COLOR\_FORMAT = 'hex'

The format in which the extracted colors will be exported (e.g.

dict EXPORT\_PALETTE\_TEMPLATES = {}

Dictionary of palette templates that can be used to organize extracted colors into palettes to export.

dict PALETTE\_COLOR\_TYPES\_CONTAINED = {}

Dictionary of the color types that are contained within each palette template.

• bool SAVE\_CHECK = False

Flag to check if user wants to save extracted color palettes.

• bool SHOW\_PREVIEW = False

Flag to show a preview of extracted palettes.

• bool ADAPTIVE\_PALETTE = False

Flag to generate 2 adaptive color palettes.

• bool MOOD\_PALETTE = False

Flag to generate 2 mood color palettes.

• bool SAVE\_RAW = False

Flag to save raw extracted colors.

• bool PASTEL\_L = False

Flag to convert light color type to pastel.

• bool PASTEL\_N = False

Flag to convert normal color type to pastel.

bool PASTEL D = False

Flag to convert dark color type to pastel.

• str LIGHT\_MOOD\_PALETTE\_NAME = 'light-mood'

The palette name of the light-themed mood palette.

• str DARK\_MOOD\_PALETTE\_NAME = 'dark-mood'

The palette name of the dark-themed mood palette.

• str LIGHT\_ADAPTIVE\_PALETTE\_NAME = 'goldilocks-light'

The palette name of the light-themed adaptive palette.

str DARK\_ADAPTIVE\_PALETTE\_NAME = 'goldilocks-dark'

The palette name of the dark-themed adaptive palette.

• dict VALID\_COLOR\_SET

A set of valid color names used to check user-defined color palettes from the configuration file.

#### 5.2.1 Function Documentation

#### check\_path()

```
check_path (
          path )
```

Check the path to make sure it exists.

#### **Parameters**

path	The path to a directory.
------	--------------------------

### Returns

True if the path exists and is not a file, False otherwise.

# check\_source()

```
check_source (
          filepath )
```

Checks to make sure the path leads to a file.

### **Parameters**

filepath	Path to file with filename and file extension.
----------	--

# Returns

True if file exists, False otherwise.

### check\_sources()

```
check_sources (
     filepaths,
     path = None )
```

Checks each of the sources provided and removes any bad sources.

Any filepaths or source files that are not images or do not exist get removed.

#### **Parameters**

filepaths	List of file paths.
path	A path to the images, if it is provided.

### Returns

True if all/some sources are good, False if all sources are bad.

# extract\_color\_palettes()

```
extract_color_palettes ( )
```

Handles color extraction from image(s).

### handle\_args()

```
handle_args ( )
```

Handles the arguments passed to PyPalEx.

# handle\_config()

```
handle_config ( )
```

Handle the PyPalEx configuration file settings.

# main()

```
main ()
```

Main script function.

# preview\_and\_save()

Shows a preview of and saves the extracted color palette(s).

### **Parameters**

extractor	An Extractor object.
save_type	A string character that specifies what type of palette to extract and save (i.e. a = adaptive, m =
	mood, r = raw, t = templates).
img_index	The integer index used to identify the image name.  Generated by Doxygen

### set\_global\_args()

```
set_global_args (
          args )
```

Sets the global variables using the arguments.

### **Parameters**

```
args User-supplied arguments.
```

### setup\_argument\_parser()

```
setup_argument_parser ( )
```

Sets up the argument parser for command line arguments.

#### Returns

A command line argument parsing object.

#### 5.2.2 Variable Documentation

### ADAPTIVE PALETTE

```
bool ADAPTIVE_PALETTE = False
```

Flag to generate 2 adaptive color palettes.

# CONFIG\_FILENAME

```
str CONFIG_FILENAME = 'palex-config.yaml'
```

Filename of the configuration file.

# DARK\_ADAPTIVE\_PALETTE\_NAME

```
str DARK_ADAPTIVE_PALETTE_NAME = 'goldilocks-dark'
```

The palette name of the dark-themed adaptive palette.

# DARK\_MOOD\_PALETTE\_NAME

```
str DARK_MOOD_PALETTE_NAME = 'dark-mood'
```

The palette name of the dark-themed mood palette.

# EXPORT\_COLOR\_FORMAT

```
str EXPORT_COLOR_FORMAT = 'hex'
```

The format in which the extracted colors will be exported (e.g.

```
'hsv', 'rgb', 'hex', 'ansi').
```

### EXPORT\_FILE\_FORMAT

```
str EXPORT_FILE_FORMAT = 'json'
```

The format of the files to be exported (e.g.

'json', 'yaml').

# EXPORT\_PALETTE\_TEMPLATES

```
dict EXPORT_PALETTE_TEMPLATES = { }
```

Dictionary of palette templates that can be used to organize extracted colors into palettes to export.

### **FILENAMES**

```
list FILENAMES = []
```

List of image filenames (contain file extensions).

# IMAGE\_NAMES

```
list IMAGE_NAMES = []
```

List of image names.

# LIGHT\_ADAPTIVE\_PALETTE\_NAME

```
str LIGHT_ADAPTIVE_PALETTE_NAME = 'goldilocks-light'
```

The palette name of the light-themed adaptive palette.

# LIGHT\_MOOD\_PALETTE\_NAME

```
str LIGHT_MOOD_PALETTE_NAME = 'light-mood'
```

The palette name of the light-themed mood palette.

### MOOD\_PALETTE

```
bool MOOD_PALETTE = False
```

Flag to generate 2 mood color palettes.

# OUTPUT\_PATH

```
str OUTPUT_PATH = ''
```

The path to the output directory where all exported files will be saved.

# PALETTE\_COLOR\_TYPES\_CONTAINED

```
dict PALETTE_COLOR_TYPES_CONTAINED = {}
```

Dictionary of the color types that are contained within each palette template.

### PASTEL\_D

```
bool PASTEL_D = False
```

Flag to convert dark color type to pastel.

### PASTEL\_L

```
bool PASTEL_L = False
```

Flag to convert light color type to pastel.

### PASTEL N

```
bool PASTEL_N = False
```

Flag to convert normal color type to pastel.

# PROPER\_IMAGES

```
list PROPER_IMAGES = []
```

List of real/existing image file path(s).

# ${\bf SAVE\_CHECK}$

```
bool SAVE_CHECK = False
```

Flag to check if user wants to save extracted color palettes.

### SAVE\_RAW

```
bool SAVE_RAW = False
```

Flag to save raw extracted colors.

### SHOW\_PREVIEW

```
bool SHOW_PREVIEW = False
```

Flag to show a preview of extracted palettes.

# VALID\_COLOR\_SET

```
dict VALID_COLOR_SET
```

## Initial value:

```
00001 = {'red', 'light red', 'dark red', 'orange', 'light orange', 'dark orange',
00002 'yellow', 'light yellow', 'dark yellow', 'chartreuse', 'light chartreuse', 'dark
chartreuse',
00003 'green', 'light green', 'dark green', 'spring', 'light spring', 'dark spring',
00004 'cyan', 'light cyan', 'dark cyan', 'azure', 'light azure', 'dark azure',
00005 'blue', 'light blue', 'dark blue', 'violet', 'light violet', 'dark violet',
00006 'magenta', 'light magenta', 'dark magenta', 'rose', 'light rose', 'dark rose'}
```

A set of valid color names used to check user-defined color palettes from the configuration file.

# 5.3 pypalex.arg\_messages Namespace Reference

### **Functions**

• bad\_source\_message ()

Generates an error message if the sources provided were not images.

• bad\_path\_message ()

Generates an error message if the directory provided is not a valid directory.

no\_args\_help\_message ()

Generates a help message if no arguments were presented.

## 5.3.1 Function Documentation

## bad\_path\_message()

```
bad_path_message ( )
```

Generates an error message if the directory provided is not a valid directory.

### Returns

The "bad directory" message.

### bad\_source\_message()

```
bad_source_message ( )
```

Generates an error message if the sources provided were not images.

#### Returns

The "bad sources" message.

#### no\_args\_help\_message()

```
no_args_help_message ( )
```

Generates a help message if no arguments were presented.

#### Returns

The "no arguments" help message.

### 5.4 pypalex.constants Namespace Reference

#### **Variables**

- list BLACK\_RGB = [0, 0, 0]
- list WHITE\_RGB = [255, 255, 255]
- list RED\_RGB = [255, 0, 0]
- list ORANGE\_RGB = [255, 153, 0]
- list YELLOW RGB = [255, 213, 0]
- list CHARTREUSE\_RGB = [191, 255, 0]
- list GREEN\_RGB = [0, 255, 0]
- list SPRING\_RGB = [0, 255, 149]
- list CYAN\_RGB = [0, 255, 255]
- list AZURE\_RGB = [0, 128, 255]
- list BLUE\_RGB = [0, 0, 255]
- list VIOLET\_RGB = [140, 0, 255]
- list MAGENTA\_RGB = [255, 0, 255]
- list ROSE\_RGB = [255, 0, 93]
- int BLACK\_HEX = 0x000000
- int WHITE\_HEX = 0xFFFFFF
- int RED\_HEX = 0xFF0000
- int ORANGE\_HEX = 0xFF9900
- int YELLOW\_HEX = 0xFFD500
- int CHARTREUSE\_HEX = 0xBFFF00
- int GREEN\_HEX = 0x00FF00
- int SPRING\_HEX = 0x00FF95
- int CYAN\_HEX = 0x00FFFF
- int AZURE HEX = 0x0080FF
- int BLUE HEX = 0x0000FF
- int VIOLET\_HEX = 0x8C00FF
- int MAGENTA\_HEX = 0xFF00FF
- int ROSE\_HEX = 0xFF005D

- int RED\_HUE = 0
- int ORANGE\_HUE = 36
- int YELLOW\_HUE = 50
- int CHARTREUSE HUE = 75
- int GREEN HUE = 120
- int SPRING HUE = 155
- int CYAN HUE = 180
- int AZURE\_HUE = 210
- int BLUE\_HUE = 240
- int VIOLET\_HUE = 273
- int MAGENTA HUE = 300
- int ROSE\_HUE = 338
- list RED\_HUE\_RANGE\_MIN = [0, 20]
- list ORANGE\_HUE\_RANGE = [20, 43]
- list YELLOW HUE RANGE = [43, 64]
- list CHARTREUSE\_HUE\_RANGE = [64, 90]
- list GREEN HUE RANGE = [90, 145]
- list SPRING HUE RANGE = [145, 170]
- list CYAN\_HUE\_RANGE = [170, 195]
- list AZURE\_HUE\_RANGE = [195, 220]
- list BLUE\_HUE\_RANGE = [220, 255]
- list VIOLET\_HUE\_RANGE = [255, 290]
- list MAGENTA HUE RANGE = [290, 325]
- list ROSE\_HUE\_RANGE = [325, 350]
- list RED HUE RANGE MAX = [350, 360]
- list BLACK\_BRIGHTNESS\_RANGE = [0.0, 35.0]
- list DARK\_BRIGHTNESS\_RANGE = [35.0, 55.0]
- list NORM\_BRIGHTNESS\_RANGE = [55.0, 80.0]
- list LIGHT\_BRIGHTNESS\_RANGE = [80.0, 100.0]
- list SATURATION\_TOLERANCE\_RANGE = [15.0, 20.0]
- list PASTEL\_SATURATION\_RANGE = [20.0, 55.0]
- list PASTEL\_BRIGHTNESS\_RANGE = [65.0, 95.0]

### 5.4.1 Variable Documentation

# AZURE\_HEX

int AZURE\_HEX =  $0 \times 0080$ FF

### AZURE\_HUE

int AZURE\_HUE = 210

# AZURE\_HUE\_RANGE

list AZURE\_HUE\_RANGE = [195, 220]

# AZURE\_RGB

list AZURE\_RGB = [0, 128, 255]

# BLACK\_BRIGHTNESS\_RANGE

```
list BLACK_BRIGHTNESS_RANGE = [0.0, 35.0]
```

# **BLACK\_HEX**

```
int BLACK_HEX = 0 \times 000000
```

# **BLACK\_RGB**

```
list BLACK_RGB = [0, 0, 0]
```

# BLUE\_HEX

```
int BLUE_HEX = 0 \times 0000 FF
```

# BLUE\_HUE

```
int BLUE_HUE = 240
```

# BLUE\_HUE\_RANGE

```
list BLUE_HUE_RANGE = [220, 255]
```

# BLUE\_RGB

```
list BLUE_RGB = [0, 0, 255]
```

## CHARTREUSE\_HEX

```
int CHARTREUSE_HEX = 0xBFFF00
```

# CHARTREUSE\_HUE

```
int CHARTREUSE_HUE = 75
```

# CHARTREUSE\_HUE\_RANGE

```
list CHARTREUSE_HUE_RANGE = [64, 90]
```

# CHARTREUSE\_RGB

```
list CHARTREUSE_RGB = [191, 255, 0]
```

# CYAN\_HEX

```
int CYAN_HEX = 0 \times 0.00FFFF
```

# CYAN\_HUE

```
int CYAN_HUE = 180
```

# CYAN\_HUE\_RANGE

```
list CYAN_HUE_RANGE = [170, 195]
```

# CYAN\_RGB

```
list CYAN_RGB = [0, 255, 255]
```

# DARK\_BRIGHTNESS\_RANGE

```
list DARK_BRIGHTNESS_RANGE = [35.0, 55.0]
```

# **GREEN\_HEX**

```
int GREEN\_HEX = 0x00FF00
```

# GREEN\_HUE

```
int GREEN\_HUE = 120
```

# GREEN\_HUE\_RANGE

```
list GREEN_HUE_RANGE = [90, 145]
```

# GREEN\_RGB

```
list GREEN_RGB = [0, 255, 0]
```

# LIGHT\_BRIGHTNESS\_RANGE

```
list LIGHT_BRIGHTNESS_RANGE = [80.0, 100.0]
```

# MAGENTA\_HEX

int MAGENTA\_HEX =  $0 \times FF00FF$ 

### MAGENTA\_HUE

int MAGENTA\_HUE = 300

# MAGENTA\_HUE\_RANGE

list MAGENTA\_HUE\_RANGE = [290, 325]

# MAGENTA\_RGB

list MAGENTA\_RGB = [255, 0, 255]

### NORM\_BRIGHTNESS\_RANGE

list NORM\_BRIGHTNESS\_RANGE = [55.0, 80.0]

# ORANGE\_HEX

int ORANGE\_HEX = 0xFF9900

# ORANGE\_HUE

int ORANGE\_HUE = 36

# ORANGE\_HUE\_RANGE

list ORANGE\_HUE\_RANGE = [20, 43]

# ORANGE\_RGB

list ORANGE\_RGB = [255, 153, 0]

# PASTEL\_BRIGHTNESS\_RANGE

```
list PASTEL_BRIGHTNESS_RANGE = [65.0, 95.0]
```

# PASTEL\_SATURATION\_RANGE

```
list PASTEL_SATURATION_RANGE = [20.0, 55.0]
```

# RED\_HEX

```
int RED_HEX = 0 \times FF0000
```

# RED\_HUE

```
int RED_HUE = 0
```

# RED\_HUE\_RANGE\_MAX

```
list RED_HUE_RANGE_MAX = [350, 360]
```

# RED\_HUE\_RANGE\_MIN

```
list RED_HUE_RANGE_MIN = [0, 20]
```

# RED\_RGB

```
list RED_RGB = [255, 0, 0]
```

# ROSE\_HEX

int ROSE\_HEX = 0xFF005D

# ROSE\_HUE

int ROSE\_HUE = 338

# ROSE\_HUE\_RANGE

list ROSE\_HUE\_RANGE = [325, 350]

# ROSE\_RGB

```
list ROSE_RGB = [255, 0, 93]
```

# SATURATION\_TOLERANCE\_RANGE

```
list SATURATION_TOLERANCE_RANGE = [15.0, 20.0]
```

# SPRING\_HEX

```
int SPRING_HEX = 0 \times 00FF95
```

# SPRING\_HUE

```
int SPRING_HUE = 155
```

# SPRING\_HUE\_RANGE

```
list SPRING_HUE_RANGE = [145, 170]
```

# SPRING\_RGB

```
list SPRING_RGB = [0, 255, 149]
```

# VIOLET\_HEX

```
int VIOLET_HEX = 0x8C00FF
```

# VIOLET\_HUE

```
int VIOLET\_HUE = 273
```

# VIOLET\_HUE\_RANGE

```
list VIOLET_HUE_RANGE = [255, 290]
```

# VIOLET\_RGB

```
list VIOLET_RGB = [140, 0, 255]
```

### WHITE\_HEX

```
int WHITE_HEX = 0xFFFFFF
```

#### WHITE\_RGB

```
list WHITE_RGB = [255, 255, 255]
```

### YELLOW\_HEX

```
int YELLOW_HEX = 0 \times FFD500
```

#### YELLOW HUE

```
int YELLOW_HUE = 50
```

#### YELLOW HUE RANGE

```
list YELLOW_HUE_RANGE = [43, 64]
```

## YELLOW\_RGB

```
list YELLOW_RGB = [255, 213, 0]
```

# 5.5 pypalex.conversion utils Namespace Reference

## **Functions**

• hsv\_to\_hex (hsv\_array)

Convert HSV array [h,s,v] to HEX string '#ffffff'.

hex\_to\_hsv (hex\_str)

Convert HEX string '#ffffff' to HSV array [h,s,v].

• hsv\_to\_ansi (hsv\_array, background=False)

Convert HSV array [h,s,v] to an ANSI color escape code string.

· ansi to hsv (ansi string)

Converts ANSI color escape code string tp HSV array [h,s,v].

hex\_to\_ansi (hex\_str, background=False)

Convert HEX string '#ffffff' to an ANSI color escape code string.

ansi\_to\_hex (ansi\_string)

Converts ANSI color escape code string to HEX string '#ffffff'.

rgb\_to\_hsv (rgb\_array)

Converts RGB array [r,g,b] to HSV array [h,s,v].

hsv\_to\_rgb (hsv\_array)

Convert HSV array [h,s,v] to RGB array [r,g,b].

rgb\_to\_hex (rgb\_array)

Convert RGB array [r,g,b] to HEX string '#ffffff'.

hex\_to\_rgb (hex\_str)

Convert HEX string '#ffffff' to RGB array [r,g,b].

• rgb\_to\_ansi (rgb\_array, background=False)

Convert RGB array [r,g,b] to an ANSI color escape code string.

• ansi\_to\_rgb (ansi\_string)

Converts ANSI color escape code string to an RGB array.

#### 5.5.1 Function Documentation

### ansi\_to\_hex()

Converts ANSI color escape code string to HEX string '#ffffff'.

ANSI where 033[38;2;r;g;bm] is for the foreground color and 033[48;2;r;g;bm] is for the background color. HEX string is in the set ["#000000", "#fffffff"].

#### **Parameters**

ansi_string	ANSI color escape code string.
-------------	--------------------------------

#### Returns

A HEX string.

### ansi\_to\_hsv()

Converts ANSI color escape code string tp HSV array [h,s,v].

ANSI where 033[38;2;r;g;bm] is for the foreground color and 033[48;2;r;g;bm] is for the background color. HSV where h is in the set [0, 359] and s, v are in the set [0.0, 100.0].

### **Parameters**

ansi_string   ANSI color escape code string	ıa.
---	-----

### Returns

HSV array [h,s,v].

### ansi\_to\_rgb()

Converts ANSI color escape code string to an RGB array.

#### Note

This function is dependent on the ANSI string to be formatted like '\033[{};2;{};{}m' or '\u001b[{};2;{};{}m' or something similar. For more information about these ANSI escape codes, here are some sources: https://en.wikipedia.org/wiki/ANSI\_escape\_code#8-bit https-://stackoverflow.com/questions/4842424/list-of-ansi-color-escape-sequences/33206814# https://stackoverflow.com/questions/45782766/color-python-output-given-rrggbb-hex-v

#### **Parameters**

ansi string   ANSI color escape code string.	ansi string	ANSI color escape code string.
--	-------------	--------------------------------

### Returns

RGB array [r,g,b].

# hex\_to\_ansi()

Convert HEX string '#ffffff' to an ANSI color escape code string.

HEX string is in the set ["#000000", "#ffffff"]. ANSI where 033[38;2;r;g;bm] is for the foreground color and 033[48;2;r;g;bm] is for the background color.

#### **Parameters**

hex_str	HEX string '#ffffff'.	
background	Flag for if the HEX string is for a background or not.	

# Returns

ANSI escape code string.

# hex\_to\_hsv()

```
\label{eq:hex_str} \begin{array}{ll} \text{hex\_to\_hsv (} \\ & \text{hex\_str )} \end{array}
```

Convert HEX string '#ffffff' to HSV array [h,s,v].

HEX string is in the set ["#000000", "#ffffff"]. HSV where h is in the set [0, 359] and s, v are in the set [0.0, 100.0].

### **Parameters**

hex_str	HEX string '#ffffff'.

### Returns

HSV array [h,s,v].

# hex\_to\_rgb()

```
hex_to_rgb (
```

```
hex_str )
```

Convert HEX string '#ffffff' to RGB array [r,g,b].

HEX string is in the set ["#000000", "#ffffff"]. RGB where [r,g,b] are in the set [0, 255].

#### **Parameters**

```
hex_str HEX string '#ffffff'.
```

#### Returns

RGB array [r,g,b].

### hsv\_to\_ansi()

```
\label{eq:hsv_array} \begin{array}{ll} \mbox{hsv\_array,} \\ \mbox{background = False )} \end{array}
```

Convert HSV array [h,s,v] to an ANSI color escape code string.

HSV where h is in the set [0, 359] and s, v are in the set [0.0, 100.0]. ANSI where 033[38;2;r;g;bm is for the foreground color and 033[48;2;r;g;bm is for the background color.

### **Parameters**

hsv_array	HSV array [h,s,v].
background	Flag for if the HSV color is for a background or not.

#### Returns

ANSI escape code string.

# hsv\_to\_hex()

Convert HSV array [h,s,v] to HEX string '#ffffff'.

HSV where h is in the set [0, 359] and s, v are in the set [0.0, 100.0]. HEX string is in the set ["#000000", "#ffffff"].

#### **Parameters**

hsv_array	HSV array [h,s,v].

#### Returns

A HEX string.

#### hsv\_to\_rgb()

Convert HSV array [h,s,v] to RGB array [r,g,b].

HSV where h is in the set [0, 359] and s, v are in the set [0.0, 100.0]. RGB where [r,g,b] are in the set [0, 255]. Formula adapted from https://www.rapidtables.com/convert/color/hsv-to-rgb.html

#### **Parameters**

```
hsv_array HSV array [h,s,v].
```

#### Returns

RGB array [r,g,b].

### rgb\_to\_ansi()

Convert RGB array [r,g,b] to an ANSI color escape code string.

An RGB [r,g,b] array is used to generate an ANSI escape code of the RGB color for use in the terminal CLI. The basic format for these codes depends on if it will be used for foreground or background color. Use 033[38;2;r;g;bm] for the foreground color. Use 033[48;2;r;g;bm] for the background color.

# Note

```
For more information about these ANSI escape codes, here are some sources: https↔://en.wikipedia.org/wiki/ANSI_escape_code#8-bit https://stackoverflow.↔
com/questions/4842424/list-of-ansi-color-escape-sequences/33206814#33206814
https://stackoverflow.com/questions/45782766/color-python-output-given-rrggbb-hex-v
```

#### **Parameters**

rgb_array	RGB array [r,g,b].
background	Flag for if the RGB color is for a background or not.

### Returns

ANSI escape code string of the RGB color.

#### rgb\_to\_hex()

Convert RGB array [r,g,b] to HEX string '#ffffff'.

RGB where [r,g,b] are in the set [0, 255]. HEX string is in the set ["#000000", "#ffffff"].

#### **Parameters**

```
rgb_array RGB array [r,g,b].
```

#### Returns

A HEX string.

### rgb\_to\_hsv()

Converts RGB array [r,g,b] to HSV array [h,s,v].

RGB where [r,g,b] are in the set [0, 255]. HSV where h is in the set [0, 359] and s, v are in the set [0.0, 100.0]. Formula adapted from https://www.rapidtables.com/convert/color/rgb-to-hsv.html

### **Parameters**

```
rgb_array RGB array [r,g,b].
```

#### Returns

HSV array [h,s,v].

### 5.6 pypalex.extraction\_utils Namespace Reference

#### **Functions**

• extract\_ratios (hsv\_img\_matrix\_2d)

Extracts the ratios of hues per pixel.

construct\_base\_color\_dictionary (hsv\_img\_matrix\_2d)

Constructs dictionary of base colors from an array of HSV pixel values.

• extract\_colors (base\_color\_dict, ratios=None)

Extracts dominant light, normal and dark colors from each of the base colors.

check\_missing\_colors (base\_color\_dict, extracted\_colors\_dict)

Checks for any missing colors in the base color dictionary and borrows them from the surrounding colors.

generate\_remaining\_colors (extracted\_colors\_dict, ratios)

Generate the remaining black and white, and background and foreground colors.

extract\_color\_types (color\_data)

Extracts the dominant color types from a base color.

• set\_missing\_color (extracted\_colors\_dict, color\_name)

Sets a specified color in the extracted colors dictionary by borrowing from other extracted colors.

get\_dominant\_hue (extracted\_colors\_dict, ratios)

Calculates the dominant hue.

generate\_black\_and\_white (dominant\_hue)

Generates black and white color types using the dominant hue.

• generate\_background\_and\_foreground (dominant\_hue, complementary\_hue)

Generates the background and foreground colors.

sort\_by\_sat\_and\_bright\_value (hsv\_base\_color\_matrix, color\_name, ratios)

Sorts the colors by their saturation and brightness values.

extract\_dominant\_color (hsv\_color\_type\_matrix)

Extracts the dominant color from a color type.

check\_missing\_color\_types (light\_color, norm\_color, dark\_color, black\_color, achromatic\_light, achromatic
 —norm, achromatic\_dark, achromatic\_black)

Checks to make sure all the color types have been properly set.

get left and right colors (origin color name, color type=")

Gets the color names of the colors that are to the left and right of the originating color.

• borrow\_color (extracted\_colors\_dict, origin, borrow\_left, borrow\_right, color\_type=")

Borrows a color from one of the extracted color types of the base colors.

get\_dominant\_color\_name (ratios)

Get the base name of the dominant color from a dictionary of ratios.

calculate\_centroid (hsv\_color\_type\_matrix)

Calculates the centroid for a color type.

find\_closest\_to\_centroid (hsv\_color\_type\_matrix, centroid)

Finds a color from a color type that is closest to the centroid.

• get\_hue\_shift\_value (hue, shift\_percentage)

Gets the appropriate percentage to shift a hue value based on the hue provided.

calculate dist between 2 colors (hsv color1, hsv color2)

Calculates the distance between 2 HSV colors.

### 5.6.1 Function Documentation

### borrow color()

Borrows a color from one of the extracted color types of the base colors.

### Parameters

extracted_colors_dict	A dictionary of extracted colors.
origin	A string that represents the name of the originating color.
borrow_left	A string that represents the name of the color to borrow from, to the left of origin.
borrow_right	A string that represents the name of the color to borrow from, to the right of origin.
color_type	A string that represents the type of color ('light', 'dark', " for normal).

#### Returns

A numpy array of a borrowed color.

#### calculate\_centroid()

Calculates the centroid for a color type.

The centroid is basically the average color of a set of colors in [h,s,v] format. The centroid is a point in 3-dimensional space. The following sources were used to make this algorithm:  $http://mkweb.bcgsc. \leftarrow ca/color-summarizer/?faq#averagehue and <math>https://stackoverflow.com/a/8170595/17047816$ 

#### **Parameters**

hsv_color_type_matrix	A 2D numpy array of a color type in [h,s,v] format.
-----------------------	---

#### Returns

List of centroid color values in [h,s,l] format.

### calculate\_dist\_between\_2\_colors()

Calculates the distance between 2 HSV colors.

#### **Parameters**

hsv_color1	A list or numpy array of a color in HSV format [h, s, v].
hsv_color2	A list or numpy array of a color in HSV format [h, s, v].

#### Returns

A float value that represents the distance between 2 colors.

## check\_missing\_color\_types()

```
achromatic_norm,
achromatic_dark,
achromatic_black )
```

Checks to make sure all the color types have been properly set.

If a color type is missing, then it will be derived from the existing color types.

#### Note

I'm using the normalization formula from https://stats.stackexchange.com/a/281164

#### **Parameters**

light_color	A numpy array of a light color type in [h,s,v] format.
norm_color	A numpy array of a normal color type in [h,s,v] format.
dark_color	A numpy array of a dark color type in [h,s,v] format.
black_color	A numpy array of a black color type in [h,s,v] format.
achromatic_light	A numpy array of an achromatic light color type in [h,s,v] format.
achromatic_norm	A numpy array of an achromatic normal color type in [h,s,v] format.
achromatic_dark	A numpy array of an achromatic dark color type in [h,s,v] format.
achromatic_black	A numpy array of an achromatic black color type in [h,s,v] format.

### check\_missing\_colors()

Checks for any missing colors in the base color dictionary and borrows them from the surrounding colors.

#### **Parameters**

base_color_dict	A dictionary of 2D numpy arrays for each of the base colors.
extracted_colors_dict	A dictionary of extracted colors.

#### construct\_base\_color\_dictionary()

```
{\it construct\_base\_color\_dictionary (} \\ {\it hsv\_img\_matrix\_2d )}
```

Constructs dictionary of base colors from an array of HSV pixel values.

Base colors are classified as [red, orange, yellow, chartreuse, green, spring, cyan, azure, blue, violet, magenta, rose].

### Note

This function operates on the assumption that all the elements in the hsv\_img\_matrix\_2d array are sorted in ascending order using the hue (h) value.

#### **Parameters**

	hsv ima matrix 2d	A 2D numpy array of pixels from an image, in [h,s,v] format.
--	-------------------	--

### Returns

Dictionary of base colors.

# extract\_color\_types()

Extracts the dominant color types from a base color.

A color type is either a light, normal or dark version of a base color.

#### **Parameters**

color_data	A tuple whose elements are a 2D numpy array of a base color, color name string and ratios	
	dictionary.	

### Returns

List of dominant color types, where each color type is a numpy array in [h,s,v] format.

# extract\_colors()

Extracts dominant light, normal and dark colors from each of the base colors.

# **Parameters**

base_color_dict	A dictionary of 2D numpy arrays for each of the base colors.
ratios	A dictionary of color ratios (percentages) in set [0.0, 100.0] for each of the base colors.

### Returns

Dictionary of light, normal and dark color types for each of the base colors.

# extract\_dominant\_color()

Extracts the dominant color from a color type.

A color type is either a light, normal, or dark version of a base color.

#### **Parameters**

hsv_color_type_matrix   A 2D numpy array of a color type where e	every element is a list in [h,s,v] format.
--	--

#### Returns

A numpy array of a dominant color in [h,s,v] format.

# extract\_ratios()

Extracts the ratios of hues per pixel.

#### **Parameters**

hsv_img_matrix_2	A 2D numpy array of pixels, where each element/pixel is a list of color values in [h,s,v]
	format.

#### Returns

Dictionary of hue ratios (percentage) in set [0.0, 100.0]

### find\_closest\_to\_centroid()

Finds a color from a color type that is closest to the centroid.

#### **Parameters**

hsv_color_type_matrix	A 2D numpy array of a color type where every element is a list in [h,s,v] format.
centroid	List of centroid color values in [h,s,l] format.

### Returns

List of all the colors in [h,s,v] format that are the shortest distance away from the centroid.

### generate\_background\_and\_foreground()

Generates the background and foreground colors.

The background and foreground colors are based on the dominant hue in an image and it's complimentary hue. The saturation and brightness values for the background and foreground colors need to be hardcoded to be easier to look at.

#### **Parameters**

dominant_hue	An integer that represents the dominant hue in an image.
complementary_hue	An integer that represents the complimentary hue to the dominant hue.

#### Returns

A list of light and dark background and foreground colors in [h,s,v] format.

### generate\_black\_and\_white()

Generates black and white color types using the dominant hue.

The saturation and brightness values, for the black and white color types, needs to be hardcoded in order to not interfere with the background and foreground colors.

#### **Parameters**

dominant_hue	An integer that represents the dominant hue in an image.
--------------	--

### Returns

List of black and white color types in [h,s,v] format.

## generate\_remaining\_colors()

Generate the remaining black and white, and background and foreground colors.

#### **Parameters**

extracted_colors_dict	A Dictionary of extracted colors.	
ratios	A Dictionary of ratios of the base colors in the image.	l

Generated by Doxygen

### get\_dominant\_color\_name()

Get the base name of the dominant color from a dictionary of ratios.

### **Parameters**

ratios	A dictionary that contains ratios for the 12 base colors as well as for each of their color types.
--------	--

### Returns

A string that represents the name of the dominant color from one of the 12 base colors.

### get\_dominant\_hue()

Calculates the dominant hue.

The dominant hue, also referred to as the average hue, is based on the color ratios and the colors extracted from an image.

### **Parameters**

extracted_colors_dict	A Dictionary of extracted colors.
ratios	A Dictionary of ratios of the colors in the image (contains base and type color ratios).

### Returns

An integer that represents the dominant hue in an image.

# get\_hue\_shift\_value()

```
get_hue_shift_value (
            hue,
            shift_percentage )
```

Gets the appropriate percentage to shift a hue value based on the hue provided.

The shift percentage tells use by how much to shift a hue value based on what range the hue is in.

### **Parameters**

hue	An integer that represents a hue value in range [0, 359].	
shift_percentage	A float that represents the percentage in range [0.0, 100.0] by which to shift hue.	

#### Returns

An integer that represents the value by which to shift the hue.

#### get\_left\_and\_right\_colors()

Gets the color names of the colors that are to the left and right of the originating color.

There are two ways to think about left and right on a color wheel: from the top of the color wheel looking at the top-most color or from the bottom of the color wheel looking at the bottom-most color. This has an effect on how we think of the linear format of the color wheel. For this package we will think about left and right colors using the former option (top of the color wheel).

#### Note

There are 12 base colors to choose from: red, orange, yellow, chartreuse, green, spring, cyan, azure, blue, violet, magenta and rose.

#### **Parameters**

origin_color_name	A string that represents the name of the originating color.
color_type	A string that represents the type of color ('light', 'dark', " for normal).

#### Returns

List of color names that are to the left and right of the originating color.

#### set missing color()

Sets a specified color in the extracted colors dictionary by borrowing from other extracted colors.

#### Note

There are 12 base colors to choose from: red, orange, yellow, chartreuse, green, spring, cyan, azure, blue, violet, magenta and rose.

#### **Parameters**

extracted_colors_dict	A dictionary of extracted colors, where each color is a list in HSV format.
color_name	A string that represents one of the 12 base colors which will be set.

### sort\_by\_sat\_and\_bright\_value()

Sorts the colors by their saturation and brightness values.

A color type is either a light, normal, dark, black or achromatic version of a base color. The ratios for the light, normal and dark color types are also calculated.

#### **Parameters**

hsv_base_color_matrix	A 2D numpy array of a base color, where each element is a list in [h,s,v] format.
color_name	A string that represents the name of a base color.
ratios	A dictionary of color ratios (percentages) in set [0.0, 100.0] for each of the base colors.

#### Returns

A list of color types, where each element is a 2D numpy array of a color type whose elements are a list in [h,s,v] format.

## 5.7 pypalex.Extractor Namespace Reference

### Classes

· class Extractor

Extracts colors given a matrix of HSV values extracted from an image.

# 5.8 pypalex.file\_utils Namespace Reference

### **Functions**

• generate\_config\_file (config\_filename)

Generates a configuration file.

- raw\_dump (extracted\_colors\_dict, image\_name, output\_path, export\_file\_format, export\_color\_format) Saves the raw extracted colors into a file.
- save\_palettes (palettes, image\_name, output\_path, export\_file\_format, export\_color\_format, palette\_color
   —types=None)

Saves the color palettes of extracted colors.

#### 5.8.1 Function Documentation

## generate\_config\_file()

Generates a configuration file.

Generates a configuration file and saves it in the default Configuration Directory for PyPalEx.

#### Note

If a file with the same name already exists, it can be overwritten.

#### **Parameters**

config filename	A string that represents the filename of the configuration file (e.g. 'p	palex-config.yaml').
		· · · · · · · · · · · · · · · · · · ·

# raw\_dump()

Saves the raw extracted colors into a file.

Note

If a file with the same name already exists, it can be overwritten.

#### **Parameters**

extracted_colors_dict	A dictionary of colors.	
image_name	A string that represents the name of the image from where the colors were extracted (e.g. 'forest_wallpaper', 'bubblegum', etc).	
output_path	A string that specifies the directory where to save the file (can be a blank string).	
export_file_format	A string that specifies the format of the file that will be exported (e.g. 'json', 'yaml').	
export_color_format	A string that specifies the format of the colors that will be exported (e.g. 'hsv', 'rgb', 'hex', 'ansi').	

# save\_palettes()

Saves the color palettes of extracted colors.

Each palette is saved to its own individual file.

### Note

If files with the same name already exist, they can be overwritten.

#### **Parameters**

palettes	A dictionary of palettes that were organized based on the palette templates.	
image_name	A string that represents the name of the image from where the colors were extracted (e.g. 'forest_wallpaper', 'bubblegum', etc).	
output_path	A string that specifies the directory where to save the file (can be a blank string).	
export_file_format	A string that specifies the format of the file that will be exported (e.g. 'json', 'yaml').	
export_color_format	A string that specifies the format of the colors that will be exported (e.g. 'hsv', 'rgb', 'hex', 'ansi').	
palette_color_types	A dictionary that holds flags (True / False) for the color types contained in each palette and if those color types are pastel or not.	

# 5.9 pypalex.image\_utils Namespace Reference

### **Functions**

• process\_image (image)

Processes PIL Image object.

• rescale\_image (image)

Rescales image to a smaller sampling size while maintaining aspect ration.

process\_helper (rgb\_matrix\_2d)

Helper function for multiprocessing conversion operations.

# 5.9.1 Function Documentation

### process\_helper()

Helper function for multiprocessing conversion operations.

Helps convert from [r,g,b] to [h,s,v].

## Parameters

rgb_matrix_2d	A 2D matrix of rgb values.

### Returns

A numpy array/2D matrix of converted [h,s,v] values.

# process\_image()

```
\begin{tabular}{ll} process\_image & ( & & image \end{tabular} \label{table_image}
```

Processes PIL Image object.

 $\begin{tabular}{ll} \textbf{Multiprocessing example from:} & https://stackoverflow.com/a/45555516 \\ \end{tabular}$ 

### **Parameters**

image	PIL Image object.
-------	-------------------

### Returns

2D numpy array of [h,s,v] arrays (pixels) from image.

## rescale image()

```
rescale_image (
          image )
```

Rescales image to a smaller sampling size while maintaining aspect ration.

#### Note

The math behind rescaling the image came from: https://math.stackexchange. $\leftarrow$  com/a/3078131

### **Parameters**

image PIL	Image object.
-----------	---------------

## Returns

Tuple of the new width and height of image.

# 5.10 pypalex.print\_utils Namespace Reference

## **Functions**

• print\_raw\_colors (extracted\_colors\_dict, color\_format, pair\_colors=False)

Prints raw extracted colors to the Terminal.

print\_palette\_preview (palettes, color\_format)

Prints the extracted colors, organized into color palettes, to the Terminal.

• print\_default\_palette\_preview (extracted\_colors\_dict, color\_format)

Prints the extracted colors, organized into default color palettes, to the Terminal.

• print\_template\_palette\_preview (extracted\_colors\_dict, palette\_templates, color\_format)

Prints the extracted colors, organized with palette templates, to the Terminal.

print\_raw\_color (color\_name, extracted\_color, rgb\_color, ansi\_color)

Prints a raw color to the Terminal screen.

• get\_rgb\_colors (extracted\_colors\_dict, color\_format)

Constructs a dictionary of colors in RGB [r,g,b] format.

get\_ansi\_color\_codes (rgb\_colors\_dict)

Constructs an ANSI escape code dictionary using a dictionary of colors in RGB [r,g,b] format.

• make\_default\_row (rgb\_row\_color, blank\_row, border\_type=None)

Creates a string that represents a default row when printing palette previews.

• make\_foreground\_row (rbg\_foreground\_color, rbg\_background\_color)

Creates a string that represents the foreground row when printing palette previews.

- make\_panes (background\_ansi\_color, standard\_ansi\_colors, intense\_ansi\_colors)
  - Creates a string that represents the 4 rows of panes when printing palette previews.
- make\_panes\_row (background\_ansi\_color, standard\_ansi\_colors, intense\_ansi\_colors, panes\_section)

Creates a string that represents a row of panes for printing palette previews.

### 5.10.1 Function Documentation

# get\_ansi\_color\_codes()

Constructs an ANSI escape code dictionary using a dictionary of colors in RGB [r,g,b] format.

### **Parameters**

rgb_colors_dict	A dictionary of colors in RGB [r,g,b] format.
-----------------	---

### Returns

A dictionary of ANSI color escape codes.

# get\_rgb\_colors()

Constructs a dictionary of colors in RGB [r,g,b] format.

## **Parameters**

extracted_colors_dict	A dictionary of colors.
color_format	A string that specifies the format of each color in the extracted colors dictionary (e.g.
	'hsv', 'rgb', 'hex', 'ansi').

## Returns

A dictionary of RGB colors.

# make\_default\_row()

```
blank_row,
border_type = None )
```

Creates a string that represents a default row when printing palette previews.

The default row can be either a blank row or a row with a specific border.

## **Parameters**

rgb_row_color	The color of the row in RGB [r,g,b] format.
blank_row	Flag that determines if this is a blank row or a border row.
border_type	A string that specifies if this row needs a border (e.g. 'top', 'bottom').

# Returns

A string that represents a default row that can be printed.

# make\_foreground\_row()

Creates a string that represents the foreground row when printing palette previews.

## **Parameters**

rbg_foreground_color	The foreground color of the row in RGB [r,g,b] format.
rbg_background_color	The background color of the row in RGB [r,g,b] format.

# Returns

A string that represents a foreground row that can be printed.

# make\_panes()

Creates a string that represents the 4 rows of panes when printing palette previews.

background_ansi_color	lor An ANSI escape code string of the background color.	
standard_ansi_colors	A list of ANSI escape code strings for standard colors.	
intense_ansi_colors	A list of ANSI escape code strings for intense colors.	

### Returns

A string that represents the 4 rows of panes that can be printed.

## make\_panes\_row()

Creates a string that represents a row of panes for printing palette previews.

### **Parameters**

background_ansi_color	An ANSI escape code string of the background color.
standard_ansi_colors	A list of ANSI escape code strings for standard colors.
intense_ansi_colors	A list of ANSI escape code strings for intense colors.
panes_section	A string that specifies which section of the panes to make (e.g. 'top', 'middle', 'bottom').

### Returns

A string that represents a row of panes that can be printed.

# print\_default\_palette\_preview()

Prints the extracted colors, organized into default color palettes, to the Terminal.

Prints a preview of the extracted colors to the user's CLI / Terminal screen, organized into default palettes and using ANSI escape codes and ASCII characters.

### Note

The CLI / Terminal needs to be able to display ASCII characters and ANSI colors for this to work. DEPRECATED function, in favor of print\_palette\_preview().

extracted_colors_dict	A dictionary of colors.
color_format	A string that specifies the format of each color in the extracted colors dictionary (e.g. 'hsv', 'rgb', 'hex', 'ansi').

# print\_palette\_preview()

Prints the extracted colors, organized into color palettes, to the Terminal.

Prints a preview of the extracted colors to the user's CLI / Terminal screen, organized into palettes and using ANSI escape codes and ASCII characters.

## Note

The CLI / Terminal needs to be able to display ASCII characters and ANSI colors for this to work.

## **Parameters**

palettes	A dictionary of color palettes, where each item (palette) is a dictionary of colors.
color_format	A string that specifies the format of each color in the extracted colors dictionary (e.g. 'hsv', 'rgb', 'hex', 'ansi').

# print\_raw\_color()

Prints a raw color to the Terminal screen.

## **Parameters**

color_name	A string name of the color to be printed (i.e. red, yellow, blue, etc.).
extracted_color	The extracted color in any color format.
rgb_color	The extracted color in RGB color format.
ansi_color	The extracted color in ANSI color format.

# print\_raw\_colors()

Prints raw extracted colors to the Terminal.

extracted_colors_dict	A dictionary of colors.
-----------------------	-------------------------

6 Class Documentation 41

### **Parameters**

color_format	A string that specifies the format of each color in the extracted colors dictionary (e.g. 'hsv', 'rgb', 'hex', 'ansi').
pair_colors	A Flag to specify whether to print colors by pairing them by color type or not.

## print\_template\_palette\_preview()

Prints the extracted colors, organized with palette templates, to the Terminal.

Prints a preview of the extracted colors to the user's CLI / Terminal screen, organized with palette templates and using ANSI escape codes and ASCII characters.

### Note

The CLI / Terminal needs to be able to display ASCII characters and ANSI colors for this to work. DEPRECATED function, in favor of print\_palette\_preview().

## **Parameters**

extracted_colors_dict	A dictionary of colors.
palette_templates	A dictionary of palette templates.
color_format	A string that specifies the format of each color in the extracted colors dictionary (e.g. 'hsv', 'rgb', 'hex', 'ansi').

# 6 Class Documentation

## 6.1 Extractor Class Reference

Extracts colors given a matrix of HSV values extracted from an image.

## **Public Member Functions**

```
    __init__ (self)
```

Extractor Constructor.

• load (self, absolute\_image\_path, image\_name=None)

Loads the Extrator class with the provided image.

• run (self)

Main method for Extractor class.

• convert\_to\_pastel (self, pastel\_light=False, pastel\_normal=False, pastel\_dark=False)

Converts the selected color types from the extracted colors to pastel.

• set\_color\_format (self, new\_color\_format, colors\_dict=None)

Sets the color format of the colors in the extracted dictionary.

• generate\_palettes (self, palette\_templates=None)

Generates palettes based on a dictionary of palette templates.

generate\_adaptive\_palettes (self, light\_palette\_name='goldilocks-light', dark\_palette\_name='goldilocks-dark')

Generates two adaptive palettes based on the dominant color of each color-pair.

generate mood palettes (self, light palette name='light-mood', dark palette name='dark-mood')

Generates two mood palettes, starting with the dominant color in the image.

organize\_extracted\_dictionary (self)

Organizes the extracted colors dictionary.

convert\_pastel\_light (self)

Converts light color type to pastel.

convert\_pastel\_normal (self)

Converts normal color type to pastel.

convert\_pastel\_dark (self)

Converts dark color type to pastel.

generate default palettes (self)

Generates a default set of palettes from the extracted colors.

• get\_goldilocks\_colorset (self, color\_name1, color\_name2, dom\_color\_name)

Retrieves a list of 3 color types.

• get\_color\_index (self, color\_name)

Gets the index of a base color as it would appear in a linear cyclical array.

get\_color\_code\_index (self, color\_name)

Gets the color code index of a base color as it would appear in a color palette.

convert\_pastel (self, hsv\_color)

Converts/normalizes HSV color to pastel.

• get\_closest\_to\_goldilocks (self, color\_name1, color\_name2, dom\_color\_name, color\_type=")

# **Public Attributes**

· hsv\_img\_matrix\_2d

A 2D numpy array of pixels from an image in [h,s,v] format.

• image\_name

The name of the image file, without any extension (e.g.

· color\_format

A string that represents the format each color should have (e.g.

· ratio dict

A dictionary that holds the ratio of base colors in an image and is used to identify the dominant color in an image.

· base\_color\_dict

A dictionary of 2D numpy arrays for each of the 6 base colors.

extracted\_colors\_dict

A dictionary of extracted colors in [h,s,v] format.

# 6.1.1 Detailed Description

Extracts colors given a matrix of HSV values extracted from an image.

## 6.1.2 Constructor & Destructor Documentation

```
__init__()
__init__ (
__self )
```

Extractor Constructor.

**Parameters** 

### 6.1.3 Member Function Documentation

# convert\_pastel()

```
convert_pastel (
          self,
          hsv_color )
```

Converts/normalizes HSV color to pastel.

For values x in range [a, b], values x can be normalized to the new range [y, z] with the following equation:  $new_x = y + ((x-a)/(b-a)) * (z-y)$ )

Note

I'm using the normalization formula from  $\verb|https://stats.stackexchange.com/a/281164| |$ 

# Parameters

self	The object pointer.
hsv_color	List HSV color to be converted to pastel.

## convert pastel dark()

```
convert_pastel_dark (
     self )
```

Converts dark color type to pastel.

### **Parameters**

self The object pointer.

# convert\_pastel\_light()

```
convert_pastel_light (
    self )
```

Converts light color type to pastel.

# **Parameters**

```
self The object pointer.
```

# convert\_pastel\_normal()

```
convert_pastel_normal (
     self )
```

Converts normal color type to pastel.

### **Parameters**

```
self The object pointer.
```

# convert\_to\_pastel()

Converts the selected color types from the extracted colors to pastel.

There are only 3 color types to choose from: light, normal, dark.

## **Parameters**

self	The object pointer.
pastel_light	Flag to convert light color types to pastel.
pastel_normal	Flag to convert normal color types to pastel.
pastel_dark	Flag to convert dark color types to pastel.

# generate\_adaptive\_palettes()

Generates two adaptive palettes based on the dominant color of each color-pair.

Color-pairs are pairs of colors that are adjacent on the color wheel. Color-pairs include [rose, red], [orange, yellow], [chartreuse, green], [spring, cyan], [azure, blue] and [violet, magenta].

## Note

The Goldilocks naming convention is a tribute to the goldilocks zone (meaning "ideal" colors).

### **Parameters**

self	The object pointer.
light_palette_name	A string representation of the light palette name.
dark_palette_name	A string representation of the dark palette name.

## Returns

A dictionary with two palettes.

# generate\_default\_palettes()

```
\label{eq:control_generate_default_palettes} \mbox{ (} \\ self \mbox{ )}
```

Generates a default set of palettes from the extracted colors.

### **Parameters**

```
self The object pointer.
```

# Returns

A dictionary of default color palettes.

# generate\_mood\_palettes()

Generates two mood palettes, starting with the dominant color in the image.

self	The object pointer.	
light_palette_name	A string representation of the light palette name.	
dark_palette_name	A string representation of the dark palette name.	

#### Returns

A dictionary with two palettes.

## generate\_palettes()

Generates palettes based on a dictionary of palette templates.

### Note

Palette templates follow a certain structure. Each palette template has a name (key) and a dictionary (value). For a more thorough explanation please refer to the Configuration File page on the PyPalEx GitHub's Wiki page: https://github.com/AlTimofeyev/pypalex/wiki/Configuration-File

### **Parameters**

self	The object pointer.
palette_templates	A dictionary of palette template dictionaries.

### Returns

A dictionary of color palettes.

## get\_closest\_to\_goldilocks()

# get\_color\_code\_index()

Gets the color code index of a base color as it would appear in a color palette.

A color code is the NAME of a color as it appears in terminal / CLI color palettes (i.e. color0, color1, color2, ..., color15). A color code index is the index of said color where it would appear in a color palette.

### Note

```
For more information about these ANSI escape codes, here are some sources: https↔://en.wikipedia.org/wiki/ANSI_escape_code#8-bit https://stackoverflow.↔
com/questions/4842424/list-of-ansi-color-escape-sequences/33206814#33206814
https://stackoverflow.com/questions/45782766/color-python-output-given-rrggbb-hex-v
```

## **Parameters**

self	The object pointer.
color_name	A string that represents the base name of a color.

### Returns

Integer value that represents the index of a color code in a palette.

# get\_color\_index()

Gets the index of a base color as it would appear in a linear cyclical array.

## Note

For example of colors on color a wheel being shown in linear cyclical format, please refer to the internal code comment block for this function.

### **Parameters**

self	The object pointer.
color_name	A string that represents the base name of a color.

## Returns

Integer value that represents the index of a base color in a cyclical array.

# get\_goldilocks\_colorset()

Retrieves a list of 3 color types.

The color types are chosen by using a color pair and the name of a dominant color.

self	The object pointer.
color_name1	A string that represents the base name of a color.
color_name2	A string that represents the base name of a color.
dom_color_name	A string that represents the base name of the dominant color in the image.

## Returns

A list of 3 color types.

# load()

Loads the Extrator class with the provided image.

## **Parameters**

self	The object pointer.
absolute_image_path	A string that represents the absolute path to an image.
image_name	A string that represents the name of the image or any name you want to provide with the current image being used.

# organize\_extracted\_dictionary()

```
\begin{tabular}{ll} \tt organize\_extracted\_dictionary & ( \\ & self & ) \end{tabular}
```

Organizes the extracted colors dictionary.

The reorganization of the extracted colors' dictionary is done so that the (key, value) pairs appear in a specific order. This will be useful if the user wants to export the raw hierarchy of the data.

# **Parameters**

self	The object pointer.

# run()

```
run ( self\ )
```

Main method for Extractor class.

Performs extraction of colors.

self	The object pointer.

# set\_color\_format()

Sets the color format of the colors in the extracted dictionary.

There are 4 color formats to choose from: hsv, rgb, hex and ansi.

### Note

There is a loss in precision when converting between color formats more than once. If you would like to convert color formats more than once while maintaining precision of the colors, please use an auxiliary dictionary to store copies of the original extracted colors before converting to a different format.

### **Parameters**

self	The object pointer.
new_color_format	A string that represents the format each color should have (e.g. 'hsv', 'rgb', 'hex', 'ansi').
colors_dict	A dictionary of color names and their color values (optional parameter and can be a palette dictionary).

## 6.1.4 Member Data Documentation

## base\_color\_dict

```
base_color_dict
```

A dictionary of 2D numpy arrays for each of the 6 base colors.

# color\_format

```
color_format
```

A string that represents the format each color should have (e.g.

```
'hsv', 'rgb', 'hex', 'ansi').
```

## extracted\_colors\_dict

```
extracted_colors_dict
```

A dictionary of extracted colors in [h,s,v] format.

# hsv\_img\_matrix\_2d

```
hsv_img_matrix_2d
```

A 2D numpy array of pixels from an image in [h,s,v] format.

# image\_name

```
image_name
```

The name of the image file, without any extension (e.g.

```
.jpg, .png, etc.).
```

# ratio\_dict

```
ratio_dict
```

A dictionary that holds the ratio of base colors in an image and is used to identify the dominant color in an image.

The documentation for this class was generated from the following file:

· Extractor.py

# 7 File Documentation

# 7.1 \_\_main\_\_.py File Reference

Main script for PyPalEx.

# **Namespaces**

- namespace pypalex
  - Python Palette Extractor: extracts color palettes from images.
- namespace pypalex.\_\_main\_\_

### **Functions**

• main ()

Main script function.

handle\_args ()

Handles the arguments passed to PyPalEx.

extract\_color\_palettes ()

Handles color extraction from image(s).

setup\_argument\_parser ()

Sets up the argument parser for command line arguments.

• check sources (filepaths, path=None)

Checks each of the sources provided and removes any bad sources.

check\_path (path)

Check the path to make sure it exists.

• handle\_config ()

Handle the PyPalEx configuration file settings.

set\_global\_args (args)

Sets the global variables using the arguments.

preview and save (extractor, save type, img index)

Shows a preview of and saves the extracted color palette(s).

• check\_source (filepath)

Checks to make sure the path leads to a file.

### **Variables**

• str CONFIG\_FILENAME = 'palex-config.yaml'

Filename of the configuration file.

• list PROPER\_IMAGES = []

List of real/existing image file path(s).

• list FILENAMES = []

List of image filenames (contain file extensions).

• list IMAGE\_NAMES = []

List of image names.

str OUTPUT\_PATH = "

The path to the output directory where all exported files will be saved.

• str EXPORT\_FILE\_FORMAT = 'json'

The format of the files to be exported (e.g.

• str EXPORT\_COLOR\_FORMAT = 'hex'

The format in which the extracted colors will be exported (e.g.

dict EXPORT\_PALETTE\_TEMPLATES = {}

Dictionary of palette templates that can be used to organize extracted colors into palettes to export.

dict PALETTE\_COLOR\_TYPES\_CONTAINED = {}

Dictionary of the color types that are contained within each palette template.

bool SAVE CHECK = False

Flag to check if user wants to save extracted color palettes.

• bool SHOW\_PREVIEW = False

Flag to show a preview of extracted palettes.

bool ADAPTIVE PALETTE = False

Flag to generate 2 adaptive color palettes.

bool MOOD\_PALETTE = False

Flag to generate 2 mood color palettes.

• bool SAVE\_RAW = False

Flag to save raw extracted colors.

• bool PASTEL L = False

Flag to convert light color type to pastel.

• bool PASTEL\_N = False

Flag to convert normal color type to pastel.

• bool PASTEL D = False

Flag to convert dark color type to pastel.

• str LIGHT\_MOOD\_PALETTE\_NAME = 'light-mood'

The palette name of the light-themed mood palette.

str DARK\_MOOD\_PALETTE\_NAME = 'dark-mood'

The palette name of the dark-themed mood palette.

• str LIGHT ADAPTIVE PALETTE NAME = 'goldilocks-light'

The palette name of the light-themed adaptive palette.

str DARK ADAPTIVE PALETTE NAME = 'goldilocks-dark'

The palette name of the dark-themed adaptive palette.

• dict VALID COLOR SET

A set of valid color names used to check user-defined color palettes from the configuration file.

# 7.1.1 Detailed Description

Main script for PyPalEx.

Used to run from the Command Line.

# 7.1.2 Author(s)

- Created by Al Timofeyev on February 2, 2022.
- · Modified by Al Timofeyev on April 21, 2022.
- Modified by Al Timofeyev on March 6, 2023.
- · Modified by Al Timofeyev on March 22, 2023.
- Modified by Al Timofeyev on March 26, 2023.
- Modified by Al Timofeyev on April 7, 2023.
- Modified by Al Timofeyev on June 10, 2024.
- Modified by Al Timofeyev on July 8, 2024.
- · Modified by Al Timofeyev on December 15, 2024.

# 7.2 arg messages.py File Reference

Archive of messages to display for arguments supplied by user.

## **Namespaces**

· namespace pypalex

Python Palette Extractor: extracts color palettes from images.

• namespace pypalex.arg\_messages

# **Functions**

• bad\_source\_message ()

Generates an error message if the sources provided were not images.

bad\_path\_message ()

Generates an error message if the directory provided is not a valid directory.

• no\_args\_help\_message ()

Generates a help message if no arguments were presented.

# 7.2.1 Detailed Description

Archive of messages to display for arguments supplied by user.

## 7.2.2 Author(s)

- · Created by Al Timofeyev on March 3, 2022.
- Modified by Al Timofeyev on April 21, 2022.
- Modified by Al Timofeyev on March 6, 2023.
- Modified by Al Timofeyev on July 8, 2024.

# 7.3 constants.py File Reference

A collection of constants for PyPalEx.

# **Namespaces**

· namespace pypalex

Python Palette Extractor: extracts color palettes from images.

• namespace pypalex.constants

### **Variables**

- list BLACK RGB = [0, 0, 0]
- list WHITE\_RGB = [255, 255, 255]
- list RED\_RGB = [255, 0, 0]
- list ORANGE RGB = [255, 153, 0]
- list YELLOW\_RGB = [255, 213, 0]
- list CHARTREUSE\_RGB = [191, 255, 0]
- list GREEN\_RGB = [0, 255, 0]
- list SPRING\_RGB = [0, 255, 149]
- list CYAN\_RGB = [0, 255, 255]
- list AZURE RGB = [0, 128, 255]
- list BLUE RGB = [0, 0, 255]
- list VIOLET\_RGB = [140, 0, 255]
- list MAGENTA RGB = [255, 0, 255]
- list ROSE\_RGB = [255, 0, 93]
- int BLACK HEX = 0x000000
- int WHITE HEX = 0xFFFFFF
- int RED HEX = 0xFF0000
- int ORANGE\_HEX = 0xFF9900
- int YELLOW\_HEX = 0xFFD500
- int CHARTREUSE\_HEX = 0xBFFF00
- int GREEN HEX = 0x00FF00
- int SPRING HEX = 0x00FF95
- int CYAN HEX = 0x00FFFF
- int AZURE HEX = 0x0080FF
- int BLUE\_HEX = 0x0000FF
- int VIOLET HEX = 0x8C00FF
- int MAGENTA HEX = 0xFF00FF
- int ROSE\_HEX = 0xFF005D
- int RED\_HUE = 0
- int ORANGE\_HUE = 36
- int YELLOW\_HUE = 50
- int CHARTREUSE\_HUE = 75
- int GREEN HUE = 120
- int SPRING HUE = 155
- int CYAN\_HUE = 180
- int AZURE HUE = 210
- int BLUE\_HUE = 240
- int VIOLET HUE = 273
- int MAGENTA\_HUE = 300
- int ROSE\_HUE = 338
- list RED\_HUE\_RANGE\_MIN = [0, 20]
- list ORANGE\_HUE\_RANGE = [20, 43]
- list YELLOW\_HUE\_RANGE = [43, 64]
- list CHARTREUSE\_HUE\_RANGE = [64, 90]
- list GREEN\_HUE\_RANGE = [90, 145]
- list SPRING\_HUE\_RANGE = [145, 170]
- list CYAN\_HUE\_RANGE = [170, 195]
- list AZURE\_HUE\_RANGE = [195, 220]
- list BLUE\_HUE\_RANGE = [220, 255]
- list VIOLET\_HUE\_RANGE = [255, 290]
- list MAGENTA\_HUE\_RANGE = [290, 325]
- list ROSE HUE RANGE = [325, 350]
- list RED\_HUE\_RANGE\_MAX = [350, 360]

- list BLACK\_BRIGHTNESS\_RANGE = [0.0, 35.0]
- list DARK\_BRIGHTNESS\_RANGE = [35.0, 55.0]
- list NORM\_BRIGHTNESS\_RANGE = [55.0, 80.0]
- list LIGHT\_BRIGHTNESS\_RANGE = [80.0, 100.0]
- list SATURATION TOLERANCE RANGE = [15.0, 20.0]
- list PASTEL\_SATURATION\_RANGE = [20.0, 55.0]
- list PASTEL\_BRIGHTNESS\_RANGE = [65.0, 95.0]

### 7.3.1 Detailed Description

A collection of constants for PyPalEx.

## 7.3.2 Author(s)

- · Created by Al Timofeyev on February 2, 2022.
- · Modified by Al Timofeyev on April 21, 2022.
- · Modified by Al Timofeyev on March 6, 2023.
- · Modified by Al Timofeyev on May 31, 2024.
- · Modified by Al Timofeyev on June 10, 2024.
- · Modified by Al Timofeyev on October 12, 2024.

# 7.4 conversion\_utils.py File Reference

Utilities for converting between RGB, HSV, HEX.

## **Namespaces**

- · namespace pypalex
  - Python Palette Extractor: extracts color palettes from images.
- namespace pypalex.conversion\_utils

### **Functions**

- hsv\_to\_hex (hsv\_array)
  - Convert HSV array [h,s,v] to HEX string '#ffffff'.
- hex\_to\_hsv (hex\_str)
  - Convert HEX string '#ffffff' to HSV array [h,s,v].
- hsv\_to\_ansi (hsv\_array, background=False)
  - Convert HSV array [h,s,v] to an ANSI color escape code string.
- ansi\_to\_hsv (ansi\_string)
  - Converts ANSI color escape code string tp HSV array [h,s,v].
- hex\_to\_ansi (hex\_str, background=False)
  - Convert HEX string '#ffffff' to an ANSI color escape code string.
- ansi to hex (ansi string)

Converts ANSI color escape code string to HEX string '#ffffff'.

• rgb\_to\_hsv (rgb\_array)

Converts RGB array [r,g,b] to HSV array [h,s,v].

hsv\_to\_rgb (hsv\_array)

Convert HSV array [h,s,v] to RGB array [r,g,b].

rgb\_to\_hex (rgb\_array)

Convert RGB array [r,g,b] to HEX string '#ffffff'.

hex\_to\_rgb (hex\_str)

Convert HEX string '#ffffff' to RGB array [r,g,b].

• rgb\_to\_ansi (rgb\_array, background=False)

Convert RGB array [r,g,b] to an ANSI color escape code string.

ansi\_to\_rgb (ansi\_string)

Converts ANSI color escape code string to an RGB array.

## 7.4.1 Detailed Description

Utilities for converting between RGB, HSV, HEX.

# 7.4.2 Author(s)

- Created by Al Timofeyev on February 2, 2022.
- · Modified by Al Timofeyev on April 21, 2022.
- · Modified by Al Timofeyev on March 6, 2023.
- Modified by Al Timofeyev on April 5, 2023.
- Modified by Al Timofeyev on July 8, 2024.
- Modified by Al Timofeyev on October 12, 2024.

# 7.5 extraction\_utils.py File Reference

Utilities for extracting colors from the image.

## **Namespaces**

namespace pypalex

Python Palette Extractor: extracts color palettes from images.

• namespace pypalex.extraction\_utils

### **Functions**

· extract\_ratios (hsv\_img\_matrix\_2d)

Extracts the ratios of hues per pixel.

construct\_base\_color\_dictionary (hsv\_img\_matrix\_2d)

Constructs dictionary of base colors from an array of HSV pixel values.

extract\_colors (base\_color\_dict, ratios=None)

Extracts dominant light, normal and dark colors from each of the base colors.

check\_missing\_colors (base\_color\_dict, extracted\_colors\_dict)

Checks for any missing colors in the base color dictionary and borrows them from the surrounding colors.

• generate\_remaining\_colors (extracted\_colors\_dict, ratios)

Generate the remaining black and white, and background and foreground colors.

extract\_color\_types (color\_data)

Extracts the dominant color types from a base color.

• set missing color (extracted colors dict, color name)

Sets a specified color in the extracted colors dictionary by borrowing from other extracted colors.

get\_dominant\_hue (extracted\_colors\_dict, ratios)

Calculates the dominant hue.

generate black and white (dominant hue)

Generates black and white color types using the dominant hue.

• generate\_background\_and\_foreground (dominant\_hue, complementary\_hue)

Generates the background and foreground colors.

• sort\_by\_sat\_and\_bright\_value (hsv\_base\_color\_matrix, color\_name, ratios)

Sorts the colors by their saturation and brightness values.

extract\_dominant\_color (hsv\_color\_type\_matrix)

Extracts the dominant color from a color type.

check\_missing\_color\_types (light\_color, norm\_color, dark\_color, black\_color, achromatic\_light, achromatic
 —norm, achromatic\_dark, achromatic\_black)

Checks to make sure all the color types have been properly set.

get left and right colors (origin color name, color type=")

Gets the color names of the colors that are to the left and right of the originating color.

borrow\_color (extracted\_colors\_dict, origin, borrow\_left, borrow\_right, color\_type=")

Borrows a color from one of the extracted color types of the base colors.

· get dominant color name (ratios)

Get the base name of the dominant color from a dictionary of ratios.

calculate\_centroid (hsv\_color\_type\_matrix)

Calculates the centroid for a color type.

find closest to centroid (hsv color type matrix, centroid)

Finds a color from a color type that is closest to the centroid.

get\_hue\_shift\_value (hue, shift\_percentage)

Gets the appropriate percentage to shift a hue value based on the hue provided.

calculate\_dist\_between\_2\_colors (hsv\_color1, hsv\_color2)

Calculates the distance between 2 HSV colors.

# 7.5.1 Detailed Description

Utilities for extracting colors from the image.

## 7.5.2 Author(s)

- · Created by Al Timofeyev on February 10, 2022.
- Modified by Al Timofeyev on April 21, 2022.
- · Modified by Al Timofeyev on March 6, 2023.
- · Modified by Al Timofeyev on March 22, 2023.
- · Modified by Al Timofeyev on April 6, 2023.
- · Modified by Al Timofeyev on May 31, 2024.
- Modified by Al Timofeyev on June 10, 2024.
- Modified by Al Timofeyev on July 8, 2024.
- · Modified by Al Timofeyev on October 12, 2024.

# 7.6 Extractor.py File Reference

Extraction utility class for extracting colors from the image.

### Classes

class Extractor

Extracts colors given a matrix of HSV values extracted from an image.

# **Namespaces**

namespace pypalex

Python Palette Extractor: extracts color palettes from images.

· namespace pypalex.Extractor

## 7.6.1 Detailed Description

Extraction utility class for extracting colors from the image.

## 7.6.2 **Author(s)**

- · Created by Al Timofeyev on February 10, 2022.
- · Modified by Al Timofeyev on April 21, 2022.
- Modified by Al Timofeyev on March 6, 2023.
- Modified by Al Timofeyev on March 22, 2023.
- Modified by Al Timofeyev on April 5, 2023.
- Modified by Al Timofeyev on June 10, 2024.
- Modified by Al Timofeyev on July 8, 2024.
- · Modified by Al Timofeyev on October 12, 2024.

# 7.7 file\_utils.py File Reference

Utilities for file handling.

## **Namespaces**

· namespace pypalex

Python Palette Extractor: extracts color palettes from images.

• namespace pypalex.file\_utils

## **Functions**

• generate\_config\_file (config\_filename)

Generates a configuration file.

- raw\_dump (extracted\_colors\_dict, image\_name, output\_path, export\_file\_format, export\_color\_format)

  Saves the raw extracted colors into a file.
- save\_palettes (palettes, image\_name, output\_path, export\_file\_format, export\_color\_format, palette\_color
   \_types=None)

Saves the color palettes of extracted colors.

## 7.7.1 Detailed Description

Utilities for file handling.

Note

Potential point for contributors to add different output saving options.

# 7.7.2 **Author(s)**

- · Created by Al Timofeyev on April 5, 2023.
- · Modified by Al Timofeyev on July 8, 2024.
- Modified by Al Timofeyev on October 12, 2024.

# 7.8 image\_utils.py File Reference

Utilities for processing image and file handling.

## **Namespaces**

namespace pypalex

Python Palette Extractor: extracts color palettes from images.

• namespace pypalex.image\_utils

### **Functions**

process\_image (image)

Processes PIL Image object.

rescale image (image)

Rescales image to a smaller sampling size while maintaining aspect ration.

process\_helper (rgb\_matrix\_2d)

Helper function for multiprocessing conversion operations.

### 7.8.1 Detailed Description

Utilities for processing image and file handling.

## 7.8.2 Author(s)

- · Created by Al Timofeyev on February 27, 2022.
- Modified by Al Timofeyev on April 21, 2022.
- · Modified by Al Timofeyev on March 6, 2023.
- Modified by Al Timofeyev on April 5, 2023.
- · Modified by Al Timofeyev on May 16, 2024.

# 7.9 print\_utils.py File Reference

Utilities for printing preview to the screen.

## **Namespaces**

· namespace pypalex

Python Palette Extractor: extracts color palettes from images.

namespace pypalex.print\_utils

### **Functions**

• print\_raw\_colors (extracted\_colors\_dict, color\_format, pair\_colors=False)

Prints raw extracted colors to the Terminal.

print\_palette\_preview (palettes, color\_format)

Prints the extracted colors, organized into color palettes, to the Terminal.

• print\_default\_palette\_preview (extracted\_colors\_dict, color\_format)

Prints the extracted colors, organized into default color palettes, to the Terminal.

• print\_template\_palette\_preview (extracted\_colors\_dict, palette\_templates, color\_format)

Prints the extracted colors, organized with palette templates, to the Terminal.

print\_raw\_color (color\_name, extracted\_color, rgb\_color, ansi\_color)

Prints a raw color to the Terminal screen.

• get rgb colors (extracted colors dict, color format)

Constructs a dictionary of colors in RGB [r,g,b] format.

• get\_ansi\_color\_codes (rgb\_colors\_dict)

Constructs an ANSI escape code dictionary using a dictionary of colors in RGB [r,g,b] format.

• make\_default\_row (rgb\_row\_color, blank\_row, border\_type=None)

Creates a string that represents a default row when printing palette previews.

make\_foreground\_row (rbg\_foreground\_color, rbg\_background\_color)

Creates a string that represents the foreground row when printing palette previews.

• make\_panes (background\_ansi\_color, standard\_ansi\_colors, intense\_ansi\_colors)

Creates a string that represents the 4 rows of panes when printing palette previews.

• make\_panes\_row (background\_ansi\_color, standard\_ansi\_colors, intense\_ansi\_colors, panes\_section)

Creates a string that represents a row of panes for printing palette previews.

## 7.9.1 Detailed Description

Utilities for printing preview to the screen.

Note

Potential point for contributors to add different printing options, maybe even a printing option that displays in a GUI.

## 7.9.2 Author(s)

- · Created by Al Timofeyev on April 5, 2023.
- · Modified by Al Timofeyev on July 8, 2024.
- · Modified by Al Timofeyev on October 12, 2024.

# Index

init	pypalex.constants, 13
Extractor, 43	check_missing_color_types
mainpy, 50	pypalex.extraction_utils, 25
	check_missing_colors
ADAPTIVE_PALETTE	pypalex.extraction_utils, 26
pypalexmain, 7	check_path
ansi_to_hex	pypalexmain, 5
pypalex.conversion_utils, 19	check_source
ansi_to_hsv	pypalexmain, 5
pypalex.conversion_utils, 19	check_sources
ansi_to_rgb	pypalexmain, 5
pypalex.conversion_utils, 19	color format
arg_messages.py, 52	Extractor, 49
AZURE_HEX	CONFIG_FILENAME
pypalex.constants, 12	pypalexmain, 7
AZURE_HUE	constants.py, 53
pypalex.constants, 12	construct_base_color_dictionary
AZURE_HUE_RANGE	pypalex.extraction_utils, 26
pypalex.constants, 12	conversion_utils.py, 55
AZURE_RGB	convert_pastel
pypalex.constants, 12	Extractor, 43
	convert pastel dark
bad_path_message	Extractor, 43
pypalex.arg_messages, 10	convert_pastel_light
bad_source_message	Extractor, 43
pypalex.arg_messages, 10	convert_pastel_normal
base_color_dict	Extractor, 44
Extractor, 49	convert_to_pastel
BLACK_BRIGHTNESS_RANGE	Extractor, 44
pypalex.constants, 12	CYAN HEX
BLACK_HEX	pypalex.constants, 14
pypalex.constants, 13	CYAN HUE
BLACK_RGB	pypalex.constants, 14
pypalex.constants, 13	CYAN_HUE_RANGE
BLUE_HEX	pypalex.constants, 14
pypalex.constants, 13	CYAN RGB
BLUE_HUE	pypalex.constants, 14
pypalex.constants, 13	pyparoxiconotarito, 11
BLUE_HUE_RANGE	DARK_ADAPTIVE_PALETTE_NAME
pypalex.constants, 13	pypalexmain, 7
BLUE_RGB	DARK BRIGHTNESS RANGE
pypalex.constants, 13	pypalex.constants, 14
borrow_color	DARK MOOD PALETTE NAME
pypalex.extraction_utils, 24	pypalexmain, 7
calculate_centroid	EXPORT_COLOR_FORMAT
pypalex.extraction_utils, 25	pypalexmain, 7
calculate_dist_between_2_colors	EXPORT_FILE_FORMAT
pypalex.extraction_utils, 25	pypalexmain, 8
CHARTREUSE HEX	EXPORT_PALETTE_TEMPLATES
pypalex.constants, 13	pypalexmain, 8
CHARTREUSE HUE	extract_color_palettes
pypalex.constants, 13	pypalexmain, 6
CHARTREUSE_HUE_RANGE	extract_color_types
pypalex.constants, 13	pypalex.extraction_utils, 27
CHARTREUSE RGB	extract_colors
<del>-</del>	5 a 5t_ 001010

pypalex.extraction_utils, 27	get_closest_to_goldilocks
extract_dominant_color	Extractor, 46
pypalex.extraction_utils, 27	get_color_code_index
extract ratios	Extractor, 46
pypalex.extraction_utils, 28	get_color_index
extracted colors dict	Extractor, 47
Extractor, 49	get_dominant_color_name
extraction_utils.py, 56	pypalex.extraction_utils, 30
Extractor, 41	get_dominant_hue
init , 43	pypalex.extraction_utils, 30
base_color_dict, 49	get_goldilocks_colorset
color_format, 49	Extractor, 47
	•
convert_pastel, 43	get_hue_shift_value
convert_pastel_dark, 43	pypalex.extraction_utils, 30
convert_pastel_light, 43	get_left_and_right_colors
convert_pastel_normal, 44	pypalex.extraction_utils, 31
convert_to_pastel, 44	get_rgb_colors
extracted_colors_dict, 49	pypalex.print_utils, 37
generate_adaptive_palettes, 44	GREEN_HEX
generate_default_palettes, 45	pypalex.constants, 14
generate_mood_palettes, 45	GREEN_HUE
generate_palettes, 46	pypalex.constants, 14
get_closest_to_goldilocks, 46	GREEN_HUE_RANGE
get_color_code_index, 46	pypalex.constants, 14
get_color_index, 47	GREEN_RGB
get_goldilocks_colorset, 47	pypalex.constants, 14
hsv_img_matrix_2d, 49	
image_name, 50	handle_args
load, 48	pypalexmain, 6
organize_extracted_dictionary, 48	handle_config
ratio_dict, 50	pypalexmain, 6
run, 48	hex_to_ansi
set_color_format, 48	pypalex.conversion_utils, 20
Extractor.py, 58	hex_to_hsv
	pypalex.conversion_utils, 20
file_utils.py, 59	hex_to_rgb
FILENAMES	pypalex.conversion_utils, 20
pypalexmain, 8	hsv_img_matrix_2d
find closest to centroid	Extractor, 49
pypalex.extraction_utils, 28	hsv_to_ansi
p)pa.o.nontaation_attio,o	pypalex.conversion_utils, 21
generate adaptive palettes	hsv_to_hex
Extractor, 44	pypalex.conversion_utils, 21
generate_background_and_foreground	hsv_to_rgb
pypalex.extraction_utils, 28	pypalex.conversion_utils, 22
generate_black_and_white	pypalex.conversion_utils, 22
pypalex.extraction utils, 29	image name
generate config file	Extractor, 50
pypalex.file_utils, 32	IMAGE NAMES
generate_default_palettes	pypalexmain, 8
Extractor, 45	image_utils.py, 59
	inage_utils.py, 39
generate_mood_palettes	LIGHT_ADAPTIVE_PALETTE_NAME
Extractor, 45	pypalexmain, 8
generate_palettes	LIGHT BRIGHTNESS RANGE
Extractor, 46	pypalex.constants, 14
generate_remaining_colors	
pypalex.extraction_utils, 29	LIGHT_MOOD_PALETTE_NAME
get_ansi_color_codes	pypalexmain, 8
pypalex.print_utils, 37	load

Extractor, 48	print_raw_color
	pypalex.print_utils, 40
MAGENTA_HEX	print_raw_colors
pypalex.constants, 15	pypalex.print_utils, 40
MAGENTA_HUE	print_template_palette_preview
pypalex.constants, 15	pypalex.print_utils, 41
MAGENTA_HUE_RANGE	print_utils.py, 60
pypalex.constants, 15	process_helper
MAGENTA_RGB	pypalex.image_utils, 34
pypalex.constants, 15	process_image
main	pypalex.image_utils, 34
pypalexmain, 6	PROPER_IMAGES
make_default_row	pypalexmain, 9
pypalex.print_utils, 37	pypalex, 3
make_foreground_row	pypalexmain, 3
pypalex.print_utils, 38	ADAPTIVE_PALETTE, 7
make_panes	check_path, 5
pypalex.print_utils, 38	check_source, 5
make_panes_row	check_sources, 5
pypalex.print_utils, 39	CONFIG_FILENAME, 7
MOOD_PALETTE	DARK_ADAPTIVE_PALETTE_NAME, 7
pypalexmain, 8	DARK_MOOD_PALETTE_NAME, 7
	EXPORT_COLOR_FORMAT, 7
no_args_help_message	EXPORT_FILE_FORMAT, 8
pypalex.arg_messages, 11	EXPORT_PALETTE_TEMPLATES, 8
NORM_BRIGHTNESS_RANGE	extract_color_palettes, 6
pypalex.constants, 15	FILENAMES, 8
OPANCE HEY	handle_args, 6
ORANGE_HEX	handle_config, 6
pypalex.constants, 15 ORANGE HUE	IMAGE_NAMES, 8
pypalex.constants, 15	LIGHT_ADAPTIVE_PALETTE_NAME, 8
ORANGE_HUE_RANGE	LIGHT_MOOD_PALETTE_NAME, 8
pypalex.constants, 15	main, 6
ORANGE RGB	MOOD_PALETTE, 8
pypalex.constants, 15	OUTPUT_PATH, 9
organize_extracted_dictionary	PALETTE_COLOR_TYPES_CONTAINED, 9
Extractor, 48	PASTEL_D, 9
OUTPUT PATH	PASTEL_L, 9
pypalexmain, 9	PASTEL_N, 9
pypalexmain, 9	preview_and_save, 6
PALETTE_COLOR_TYPES_CONTAINED	PROPER_IMAGES, 9
pypalexmain, 9	SAVE_CHECK, 9
PASTEL_BRIGHTNESS_RANGE	SAVE_RAW, 9
pypalex.constants, 15	set_global_args, 7
PASTEL D	setup_argument_parser, 7
pypalexmain, 9	SHOW_PREVIEW, 10
PASTEL L	VALID_COLOR_SET, 10
pypalexmain, 9	pypalex.arg_messages, 10
PASTEL N	bad_path_message, 10
pypalexmain, 9	bad_source_message, 10
PASTEL_SATURATION_RANGE	no_args_help_message, 11
pypalex.constants, 16	pypalex.constants, 11
preview_and_save	AZURE_HEX, 12
pypalexmain, 6	AZURE_HUE, 12
print_default_palette_preview	AZURE_HUE_RANGE, 12
pypalex.print_utils, 39	AZURE_RGB, 12
print_palette_preview	BLACK_BRIGHTNESS_RANGE, 12
pypalex.print_utils, 39	BLACK_HEX, 13
· · · · · · · · · · · · · · · · · · ·	

BLACK RGB, 13	hex_to_ansi, 20
BLUE HEX, 13	hex_to_hsv, 20
BLUE_HUE, 13	hex_to_rgb, 20
BLUE HUE RANGE, 13	hsv_to_ansi, 21
BLUE RGB, 13	hsv_to_hex, 21
CHARTREUSE HEX, 13	hsv_to_rgb, 22
<del>-</del> ·	— — <del>-</del>
CHARTREUSE_HUE, 13	rgb_to_ansi, 22
CHARTREUSE_HUE_RANGE, 13	rgb_to_hex, 22
CHARTREUSE_RGB, 13	rgb_to_hsv, 23
CYAN_HEX, 14	pypalex.extraction_utils, 23
CYAN_HUE, 14	borrow_color, 24
CYAN_HUE_RANGE, 14	calculate_centroid, 25
CYAN_RGB, 14	calculate_dist_between_2_colors, 25
DARK_BRIGHTNESS_RANGE, 14	check_missing_color_types, 25
GREEN_HEX, 14	check_missing_colors, 26
GREEN_HUE, 14	construct_base_color_dictionary, 26
GREEN_HUE_RANGE, 14	extract_color_types, 27
GREEN_RGB, 14	extract_colors, 27
LIGHT_BRIGHTNESS_RANGE, 14	extract_dominant_color, 27
MAGENTA HEX, 15	extract ratios, 28
MAGENTA HUE, 15	find_closest_to_centroid, 28
MAGENTA HUE RANGE, 15	generate_background_and_foreground, 28
MAGENTA_RGB, 15	generate black and white, 29
NORM_BRIGHTNESS_RANGE, 15	generate_remaining_colors, 29
ORANGE HEX, 15	get_dominant_color_name, 30
ORANGE HUE, 15	get_dominant_hue, 30
<del>-</del> '	<del>-</del>
ORANGE_HUE_RANGE, 15	get_hue_shift_value, 30
ORANGE_RGB, 15	get_left_and_right_colors, 31
PASTEL_BRIGHTNESS_RANGE, 15	set_missing_color, 31
PASTEL_SATURATION_RANGE, 16	sort_by_sat_and_bright_value, 31
RED_HEX, 16	pypalex.Extractor, 32
RED_HUE, 16	pypalex.file_utils, 32
RED_HUE_RANGE_MAX, 16	generate_config_file, 32
RED_HUE_RANGE_MIN, 16	raw_dump, 33
RED_RGB, 16	save_palettes, 33
ROSE_HEX, 16	pypalex.image_utils, 34
ROSE_HUE, 16	process_helper, 34
ROSE_HUE_RANGE, 16	process_image, 34
ROSE RGB, 16	rescale_image, 36
SATURATION_TOLERANCE_RANGE, 17	pypalex.print_utils, 36
SPRING HEX, 17	get_ansi_color_codes, 37
SPRING HUE, 17	get_rgb_colors, 37
SPRING_HUE_RANGE, 17	make default row, 37
SPRING RGB, 17	make foreground row, 38
<del>-</del>	
VIOLET_HEX, 17	make_panes, 38
VIOLET_HUE, 17	make_panes_row, 39
VIOLET_HUE_RANGE, 17	print_default_palette_preview, 39
VIOLET_RGB, 17	print_palette_preview, 39
WHITE_HEX, 17	print_raw_color, 40
WHITE_RGB, 18	print_raw_colors, 40
YELLOW_HEX, 18	print_template_palette_preview, 41
YELLOW_HUE, 18	PyPalEx: The Python Palette Extractor, 1
YELLOW_HUE_RANGE, 18	
YELLOW_RGB, 18	ratio_dict
pypalex.conversion_utils, 18	Extractor, 50
ansi_to_hex, 19	raw_dump
ansi_to_hsv, 19	pypalex.file_utils, 33
ansi_to_rgb, 19	RED_HEX
·· ·· • · · ·	pypalex.constants, 16

RED_HUE	pypalex.constants, 17
pypalex.constants, 16	VIOLET_HUE
RED_HUE_RANGE_MAX	pypalex.constants, 17
pypalex.constants, 16	VIOLET_HUE_RANGE
RED_HUE_RANGE_MIN	pypalex.constants, 17
pypalex.constants, 16	VIOLET_RGB
RED_RGB	pypalex.constants, 17
pypalex.constants, 16	WHITE HEX
rescale_image pypalex.image_utils, 36	pypalex.constants, 17
rgb_to_ansi	WHITE RGB
pypalex.conversion_utils, 22	pypalex.constants, 18
rgb_to_hex	171
pypalex.conversion_utils, 22	YELLOW_HEX
rgb to hsv	pypalex.constants, 18
pypalex.conversion_utils, 23	YELLOW_HUE
ROSE HEX	pypalex.constants, 18
pypalex.constants, 16	YELLOW_HUE_RANGE
ROSE HUE	pypalex.constants, 18
pypalex.constants, 16	YELLOW_RGB
ROSE_HUE_RANGE	pypalex.constants, 18
pypalex.constants, 16	
ROSE_RGB	
pypalex.constants, 16	
run	
Extractor, 48	
CATURATION TO ERANGE RANGE	
SATURATION_TOLERANCE_RANGE	
pypalex.constants, 17	
SAVE_CHECK	
pypalexmain, 9 save_palettes	
pypalex.file utils, 33	
SAVE RAW	
pypalex. main , 9	
set_color_format	
Extractor, 48	
set_global_args	
pypalexmain, 7	
set_missing_color	
pypalex.extraction_utils, 31	
setup_argument_parser	
pypalexmain, 7	
SHOW_PREVIEW	
pypalexmain, 10	
sort_by_sat_and_bright_value	
pypalex.extraction_utils, 31	
SPRING_HEX	
pypalex.constants, 17	
SPRING_HUE	
pypalex.constants, 17	
SPRING_HUE_RANGE pypalex.constants, 17	
SPRING RGB	
pypalex.constants, 17	
pypaion.cometante, 17	
VALID_COLOR_SET	
pypalexmain, 10	
VIOLET_HEX	